Monitoring a Cohort of Blood Donors and Recipients Using SAS/AF® FRAME

Monique Bryher, MRI Consulting, Inc., Los Angeles, CA
for the Transfusion-Transmitted Viruses Study

ABSTRACT

The Transfusion-Transmitted Virus Study (TTVS) was a National Heart Lung Blood Institute (NHLBI) sponsored investigation during the 1970s of characteristics of blood donors and recipients in four geographic areas of the United States. Its immediate purpose was to determine the incidence of transfusion-transmitted hepatitis viruses during that time. Another major purpose was to develop a long-term database and serum repository to be used subsequently to answer questions for agents for which there was no method of testing at that time.

The original database was stored in SAS, so that any investigator who wanted to use the information either had to be able to program in SAS or obtain the assistance of a SAS programmer. With the development over the past decade of assays for many new hepatitis and other viruses, a large amount of new data were added to the database. In addition, many questions about these involved choosing subsets of donors and recipients with particular characteristics.

As a result, the study found a need for an investigator unfamiliar with SAS programming to have methods of easily accessing the data, an ability to retrieve information quickly, to perform simple statistical procedures, and be able to depict the data in report and graphical formats. Because it could satisfy those complex requirements, the SAS System, Version 6.12 (Windows NT), was selected as the development tool for the TTVS database. This presentation will demonstrate in tandem how AF/FRAME and Screen Control Language (SCL) were employed to build a user-friendly system.

INTRODUCTION

Organizing data into a self-contained menu-driven system that is Windows-based is a departure from the traditional approach to managing data in academic research that has been publicly financed. In the traditional scenario, an investigator typically would meet with statisticians and programmers to determine the requirements for gathering, arranging and analyzing the data for a proposed study. There are a number of inefficiencies and disadvantages inherent in such a strategy, among them:

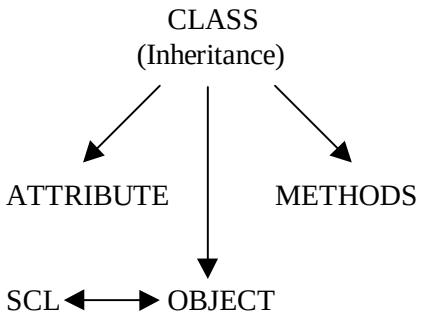
- Dependence of the investigator on programming staff to create and run reports, even if changes to existing reports are minor;
- “Accidents” occurring from storing multiple “versions of the truth”, i.e., programmers keeping differing copies of data and programs in their personal libraries;
- Researchers are often unsophisticated as to what tools are available to organize the data for presentation;
- Potential delays in receiving reports due either to other demands on programmers’ time or in miscommunications between the investigator and the programmers as to what is needed;
- Necessity in maintaining a programming staff; delays resulting from staff turnover, possibly resulting in fragmented documentation.

With advances in technology, as illustrated in the TTVS model, time is now better spent on developing a coherent system in which all of the data are available “under one roof”. The investigator need not be concerned about the organization or location of the files, or even the variable names, which are readily available by point-and-click methods and on-line help documentation.

BASIC SAS/AF FRAME CONCEPTS

FRAME is a technology designed by SAS Institute, Inc., which assists applications developers in constructing an interactive environment between the

user and his/her data. A FRAME application guides the user through graphic-user interface (GUI) based menus and windows that are filled with various *objects*. Objects are a specific representation of a *class*, which serves as the template for the object and is the foundation of object-oriented methodology. Objects inherit *instance variables* (characteristics) from their parent class, which are defined through attribute windows. They further inherit the *methods* of that class, which are executable operations, which can be defined and implemented within SCL routines.



While SAS/AF contains many classes of objects, it is possible to design applications that use only a small subset of what is available. TTVS, for instance, was built primarily using the object categories described below:

- Image Icons are visual depictions of tasks or actions to be performed. SAS/AF software supplies a library of icons, the SAS icon being the most well-known.
- The List Box class displays a box-shaped object containing a list of scrollable items from which the user makes a selection.
- The Extended Table class is similar to a List Box, but is a tabular display format for creating selection lists.
- The Text Label class is used to display messages only. It does not allow user input.
- The Container Box class groups various objects together, such as Input Fields, Push-buttons, and Graphics objects within a border.
- The Input Field class allows user input, which is then processed by the SCL.
- Push-buttons are text-labeled items that perform an action, such as displaying a list of choices or exiting a window.
- Data tables are used for either displaying or editing SAS datasets.

TTVS DATA ORGANIZATION

In TTVS, the existing files underwent a review in conjunction with the requirements of the target user community: investigators needed to be able to browse the data and build their own selection criteria and run simple statistics and reports to help them determine which recipients and donors in the repository were of interest. Because of the nature of the data, it was decided that a relational, query-driven system was preferable to the drill-down approach found in an EIS.

Because the goal of TTVS is to assist investigators in characterizing the transmission of infectious agents from donors to their recipients, the resulting database organization revolves around those two categories. The master files for donors and recipients (including controls) contain primarily demographic and medical history data, while the auxiliary files contain information with respect to the serologic and other tests that were performed and the outcomes.

All the above files were organized and indexed into the metabase (Figure 1).

Figure 1: TTVS Metabase

The TTVS metabase is a composite file whose purpose is to store information about each of the data sets found in TTVS: it contains data set locations for each of the variables, label names, formats, etc. The metabase is an essential design feature of the system, as it is accessed frequently by SCL to determine the

locations and attributes of variables selected by the investigators.

The inventory files contain the actual repository information for donors and recipients, including specimen and aliquot quantities and warehouse storage location of the specimens. These files are stored apart from the metabase and in fact, this module is not visible to the investigator; only the repository contractor has the ability to access it.

A BRIEF TOUR OF THE QUERY MODULE

Before requesting samples from the repository, the investigator uses the Query module of TTVS to scan the database of recipients (subjects) and donors for cases matching the characteristics s/he wants to study. The entry point for initiating a query is by pressing the "Explore Data" icon from the main menu (Figure 2), which opens the Data Exploration frame (Figure 3).

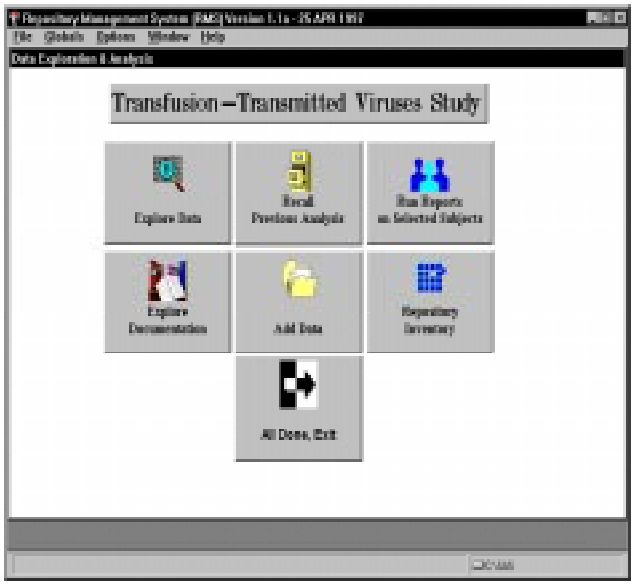


Figure 2: TTVS Main Menu

The Data Exploration frame was designed so that the investigator would be able to construct a query on one "page", i.e., be able to view all the building blocks: (1) data category, (2) variable selection, and (3) subsetting criteria simultaneously. It contains one list box, an extended table, an extended text entry, and three image icon objects.

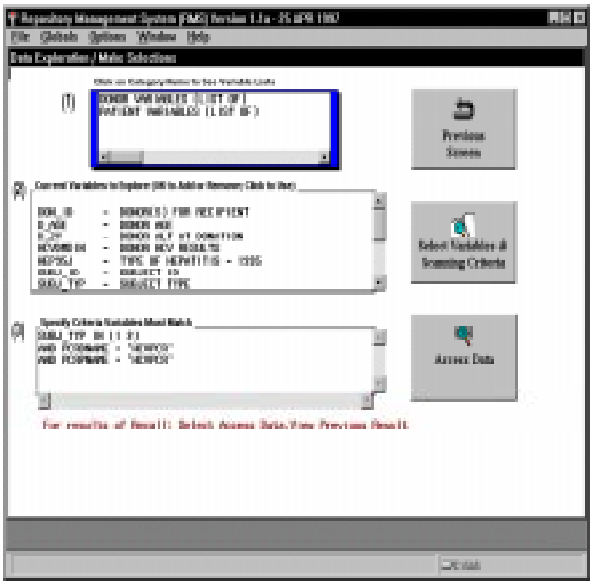


Figure 3: Data Exploration window

The sample query we will construct in our tour answers the following question:

> "Who are the subjects who sero-converted to hepatitis C virus (HCV) and who are their HCV-positive donors?"

Window (1), the list box, arranges the variables according to their major data category, i.e., patients or donors. Clicking on either choice presents the data clustered into research-coherent groupings. For example, under the Donor Variables selection, you can view and select variables arranged by demographics (Donor Characteristics), examination results (Donor Follow-up), and clinical tests and results (Donor Lab), which are then loaded into Window (2). It is not necessary to know the locations in which these variables are found, as the Screen Control Language behind the frame inserts the selected variables into SCL lists and uses those lists to point to the correct file location of those variables in the metabase. A partial listing of the code serves as an example:

```
* set-up variable lists *;
 wherep  = makelist();
 wherept = makelist();
 whered  = makelist();
 wheredn = makelist();
 wherer1 = makelist();
 wherer3 = makelist();
 rc = clearlist(wherep);
```

(continued next page)

```
 * point to the metabase *;
 do I = 1 to listlen;
    dstype=upcase(getnitemc(glist,"dstype",i));
    dsname=upcase(getnitemc(glist,"dsname",i));
    end;

 * load list by variable location *;
 if dsname = 'TTVSPPCR' then do;
     wherer1 = insertn(wherer1,i,-1);
     wherept = insertn(wherept,i,-1); end;
 else if dsname = 'TTVSDPCR' then do;
     wherer3 = insertn(wherer3,i,-1);
     wheredn = insertn(wheredn,i,-1); end;
 else if dstype = 'P' then do;
     wherep = insertn(wherep,i,-1);
     wherept = insertn(wherept,i,-1); end;
 else if dstype = 'D' then do;
     whered = insertn(whered,i,-1);
     wheredn = insertn(wheredn,i,-1); end;

* get length of each list *;
listp  = listlen(wherep);
listpt = listlen(wherept);
listd  = listlen(whered);
listdn = listlen(wheredn);
listr1 = listlen(wherer1);
listr3 = listlen(wherer3);

** build PTS dataset **;
asid = open('all');
rc   = where(asid,"upcase(DSNAME)='ALLOBS'
       and VAR ne 'SUBJ_ID'");
if attrn(asid,'any') then pkeep=1;
else pkeep=0;
rc = close(asid);
if listp>0 or pkeep=1 then do;
   submit;
     DATA PTS;
      SET DATA.ALLOBS (KEEP=SUBJ_ID VISITNO
   endsubmit;
   xwhere=wherep;
   xlist=listp;
   link blocka;
   ** get keep vars not in query **;
   asid = open('all');
   rc=where(asid,"upcase(DSNAME)='ALLOBS'");
   link blockb;
   end;
```

Once the list of variables is complete, the next step is to build the query itself. Clicking on a variable in Window (2) and then the “Select Variables and Scanning Criteria” icon object leads us step-by-step through the process of subsetting the data set by displaying a series of pop-up windows. Options to run simple statistics, such as frequencies, are available if the desired variable values/limits are not already known. Alternatively, the investigator can click on the TTVS Help option to obtain definitions of a variable and its values (Figure 4).

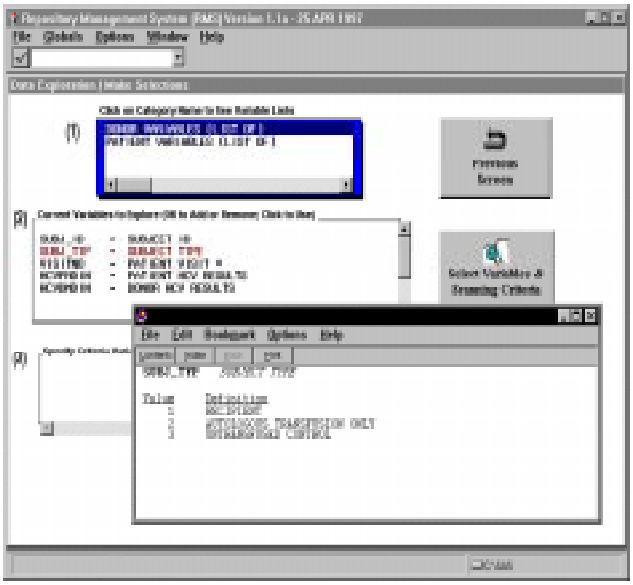


Figure 4: Variable Help

There is also a feature that displays a pop-up list of existing variable values (PROC FREQ), from which the investigator can make one or more selections (Figure 5).

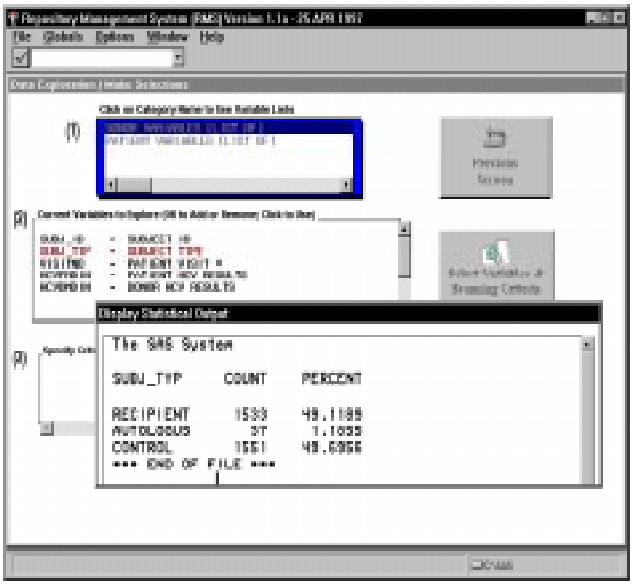


Figure 5: Select from Existing Values

After the query is complete, it is submitted by pressing “Scan/Explore Data Now” option found under the “Access Data” icon object. Additional options include deleting the selection criteria or saving the query for future retrieval.

Figure 6 displays the results of our query in a data table.

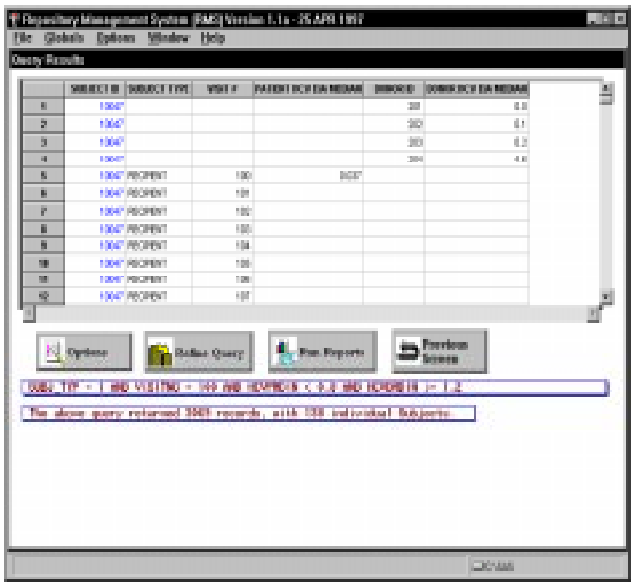


Figure 6: Query Results window

Data tables, available since Version 6.11, are objects which present data in a manner similar to a spreadsheet. Since they can be acted upon and customized using SCL methods, they are an important improvement over a static display such as PROC PRINT or PROC REPORT. An example would be dragging the mouse vertically along the right-hand side of the data table in order to scroll through the data set. The order of the columns can also be easily rearranged in a data table, or a contiguous range of rows/cells highlighted for further action.

Clicking on the Options icon object in Figure 6 displays a pop-up menu offering choices such as the ability to change the sort order of the data set in the table, delete columns, or save the query for rerunning later. By selecting the Run Reports icon, you can run standard reports, basic statistics, or invoke SAS INSIGHT in order to create simple graphs.

CONCLUSIONS

Developing a GUI-based application that allows investigators to query data in a repository and then request samples is a technological advance in the science of repository management and boasts a number of efficiencies:

- The investigator navigates through the windows and menus by pressing icons and performing actions on the data based on lists. The visual interface replaces accessing complex data sets and eliminates the necessity of keeping notebooks containing data set names and variables. Also, there is no danger of accessing the wrong version of a data set.
- Learning is done at the pace of the researcher, and the learning curve is rapid. Variables are selected by the point-and-click of a mouse, and there is a Help menu available to view the meaning of the variable names and the interpretation of their values, where appropriate.
- Once set up, the researcher is able to save the queries and recall them for future refinements and runs.
- The researcher has a turn-key, ready-to-use system that facilitates an overall view of the data, rather than having to deal with a number of data sets that have to be merged in order to obtain results or having to rely on a programmer to do the same.
- The focus of the system has now turned to the meaning of the data, rather than the method of accessing it, resulting in savings of time and increases in productivity.
- Studies that utilize data organized into a repository are not dependent on the continued participation of the original investigators so that other researchers can access the repository. The data, in essence, is preserved intact for future use.

ACKNOWLEDGEMENTS

This specific SAS/AF application was produced with funds received from NHLBI contract NO1-HB-97074. The author wishes to thank Drs. James Mosley and Eva Operskalski for granting permission to demonstrate TTVS in this paper and presentation. Thanks also to Fred Newman for his suggestions and guidance.

REFERENCES

SAS Institute, Inc. (1993) SAS AF Software: FRAME Entry, Usage and Reference, Version 6.

SAS Institute, Inc. (1994) SAS Screen Control Language: Reference, Version 6.

SAS Institute, Inc. (1995) SAS/AF Software: FRAME Class Dictionary, Version 6.

AUTHOR'S ADDRESS

Monique Bryher
MRI Consulting, Inc.
6043 Shirley Avenue
Tarzana, CA 91356
(818) 774-0043
rebwest@aol.com